UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Diseño y construcción de un móvil autoguiado para teleoperación mediante bluetooth basado en la CPU de Arduino y desarrollo de pruebas de validación

Escuela Técnica Superior de Ingeniería del Diseño

Autor: Rubén Monzó Más

Director: Luis Ignacio Gracia Calandin

Escuela Técnica Superior de Ingeniería del Diseño

Universidad Politécnica de Valencia

Septiembre 2016

DOCUMENTOS CONTENIDOS EN EL TFG

- DOCUMENTO I MEMORIA DESCRIPTIVA.
- DOCUMENTO II PRESUPUESTO.
- DOCUMENTO III PLANOS.
- DOCUMENTO IV PLIEGO DE CONDICIONES.

1 ÍNDICE DE LA MEMORIA DESCRIPTIVA

2 ÍNDICE DEL PRESUPUESTO

3 ÍNDICE DE PLANOS

3.1 Plano de dimensiones ARDUINO

3.2 Esquema de Conexiones

4 ÍNDICE DEL PLIEGO DE CONDICIONES

4.1 Condiciones Generales y Económicas

4.1.1 Condiciones generales

4.1.2 Condiciones económicas

4.2 Condiciones Técnicas y Particulares

4.2.1 Hardware

4.2.2 Software

4.2.3 Normas de Calidad

4.2.4 NNormas de Seguridad e Higiene

4.2.5 Vida útil del producto

ÍNDICE DE FIGURAS DE LA MEMORIA DESCRIPTIVA

DOCUMENTO I

MEMORIA DESCRIPTIVA

1 Introducción

En este primer capítulo se introducirá de forma genérica el trabajo llevado a cabo a modo de introducción del presente documento. Así pues se tratará el objetivo del proyecto, los antecedentes, y por último una breve estructuración de la memoria.

1.1 Objeto del trabajo.

El objetivo del presente Trabajo Final de Grado es, como el propio nombre indica, la realización tanto del diseño como la construcción de un móvil autoguiado a través de un dispositivo con bluetooth haciendo uso de una CPU de Arduino, así como llevar a cabo distintas pruebas de validación. Con este proyecto se busca, a nivel mucho más cotidiano, simular de forma experimental las pruebas que se pueden realizar con distintos móviles en terrenos de difícil acceso para personal humano.

1.2 Antecedentes.

La ingeniería de sistemas y automática es una ciencia interdisciplinar relacionada con muchos otros campos, principalmente las matemáticas, la física, la electrónica y la informática. Se encarga principalmente de la concepción y desarrollo de sistemas de tal formas que éstos funcionen de forma autónoma, con poca o ninguna intervención humana. Las aplicaciones de la ingeniería de sistemas y automática son de lo más variado: desde tecnologías de fabricación, monitorización y supervisión de procesos, robótica hasta economía y sociología. Aplicaciones típicas son, por ejemplo, el piloto automático de los aviones y barcos o el ABS de los automóviles.

Su posicionamiento como área científica y tecnología horizontal hace que la ingeniería de sistemas y automática sea vital para solucionar problemas de automatización, control y regulación de procesos en las siguientes áreas tan dispares como: electrónica y electricidad, ingeniería química, ingeniería mecánica, automoción, aeronáutica y aeroespacial, bioinformática, etc.

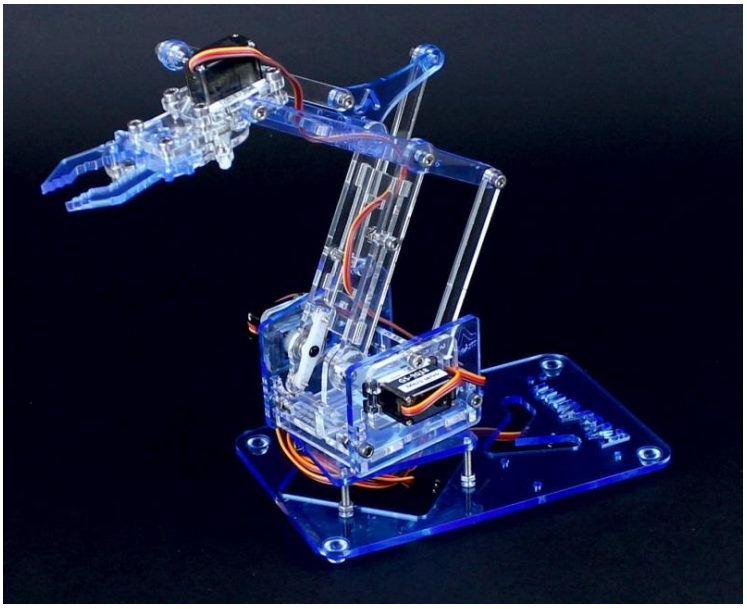


Figura 1.1 *Brazo robot implementado con Arduino.*

1.3 Estructura del Documento

El presente documento queda dividido en 7 capítulos y 1 anexo como se detalla a continuación:

- Capítulo 1: Introducción

De carácter introductorio se explica de forma general cuál es el objetivo del proyecto mediante una breve descripción, los antecedentes relacionados con la temática del documento, y por último se presenta la estructura de la memoria citando sus partes y explicando brevemente su contenido.

- Capítulo 2: Descripción del móvil y sus elementos

En este apartado se realizará una descripción de todos los elementos que forma la unidad móvil autoguiada, divididos en dos apartados: por un lado el hardware y por otro, el software utilizado.

- Capítulo 3: Planteamiento del Problema a Resolver

Se describe de forma detallada cuál es el problema que se quiere abordar.

- Capítulo 4: Análisis de las Alternativas

Durante este capítulo se plantean las posibles soluciones al problema que se ha descrito en el capítulo anterior, detallando cada opción de mejora, para por último poder hacer una priorización de las alternativas.

- Capítulo 5: Descripción de la Solución

En este capítulo se realiza el diseño y desarrollo de la solución seleccionada dentro de todas las alternativas anteriormente descritas, para poder implantarla y así solucionar el problema inicial.

- Capítulo 6: Conclusiones

Aquí se presentan las conclusiones a las que se ha llegado durante el desarrollo del proyecto, y también las líneas futuras de mejora que puedan existir.

- Capítulo 7: Referencias Bibliográficas

- Capítulo 8: Anexos

En este apartado se encuentran los Datasheet de los distintos componentes que se han usado, y el código Wiring elaborado del programa del CPU Arduino.

2 Descripción del móvil y sus elementos

2.1 Introducción.

En el presente capítulo se detallan todos los elementos por los que está formado el sistema, tanto a nivel de hardware como de software, para así poder entender el procedimiento que se ha llevado a cabo en este proyecto.

2.2 Descripción del Hardware utilizado

El móvil autoguiado en el que se basa este proyecto está constituido fundamentalmente por la CPU Arduino (aparte de todos los componentes extras ajenos a éste).

Arduino (en EEUU, Genuino para el resto del mundo) es una empresa líder mundial en el sector de software de código abierto y de hardware limpio para el ecosistema. Por esto mismo es tan utilizado en el ámbito de la electrónica y de la electricidad.

Dentro de su oferta, dispone de distintos microprocesadores perfectos para adaptarse a las distintas necesidades que se dan dentro de una empresa. Cuenta con un total de 4 microprocesadores distintos, así como distintos módulos y shields con los que se obtienen infinidad de posibilidades.

Figura 2.1 *Uno de los múltiples procesadores de Arduino.*

2.2.1 Microcontrolador Arduino UNO

El Arduino Uno es el procesador recomendado para realizar cualquier tipo de proyecto en el que se haga uso de un microcontrolador de este estilo.

Figura 2.2 *Microcontrolador Arduino UNO*

Posee tanto un puerto USB, fuente de alimentación externa, como pines de 3,3V y 5V para poder alimentarlo para su funcionamiento. Si bien admite un rango amplio de tensiones, se recomienda que se alimente con una tensión de entre 7-12V para su óptimo funcionamiento, aunque todo esto está debidamente detallado en su Datasheet.

Technical specs

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Figura 2.3 *Especificaciones técnicas del Arduino UNO.*

Sin entrar demasiado en materia de cómo funciona internamente el microprocesador, se resaltarán las partes más importantes del mismo y las que son cruciales para llevar a cabo este proyecto.

Estos elementos claves son:

- (1) Microcontrolador: es el corazón de nuestra placa, y el que se encarga de que todo funcione correctamente así como de transmitir las órdenes que le asignemos.
- (2) Entradas/Salidas digitales: en este caso hacemos usos de los pines 0 y 1, que corresponden a los pines de Transmisión (Tx) y Recepción (Rx), con los cuales nos comunicaremos con el módulo bluetooth; y de los pines que incorporan un (~) junto a su número, los cuales nos proporcionan la regulación por ancho de pulso, o PWM por sus siglas en inglés.

Figura 2.4 *Elementos claves Arduino UNO.*

2.2.2 Módulo Bluetooth HC-06

Este módulo es el que se encarga de posibilitarnos la comunicación entre nuestro controlador Arduino y cualquier dispositivo con bluetooth que utilizaremos como dispositivo de control.

Posee 2 leds por la cara donde se sitúa el circuito impreso que nos informan de manera visual si está conectado a cualquier tipo de tensión, así de si hay algún dispositivo bluetooth conectado a él.

Figura 2.5 *Módulo Bluetooth HC-06.*

2.2.3 Circuito Integrado L293D

El circuito integrado L293D, el cual está compuesto por 16 patas, incluye cuatro circuitos internos ideales para manejar cargas de potencia media, especialmente pequeños motores o cargas inductivas. Tiene la capacidad de controlar corriente de hasta 600mA por circuito y una tensión que oscila en el rango de 4,5V hasta los 36V.

La idoneidad para este proyecto del L293D es que cada circuito es capaz de configurar la mitad de un puente H, con lo cual combinando los 4 circuitos podemos obtener 2 puentes H completos, los cuales permitirán el manejo de los 2 motores que posee nuestro móvil.

Esto nos proporcionará un manejo bidireccional, con un frenado rápido y la posibilidad de implementar sin dificultad un control de velocidad.

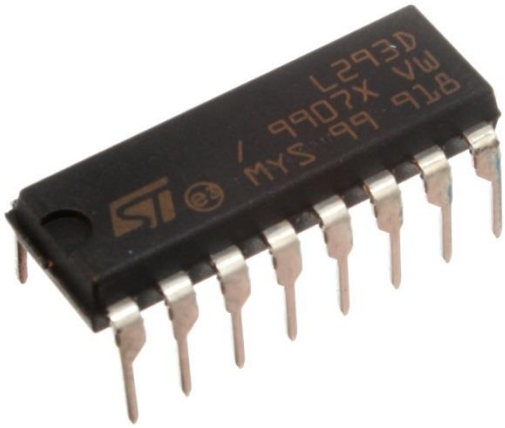


Figura 2.6 *Circuito Integrado L293D.*

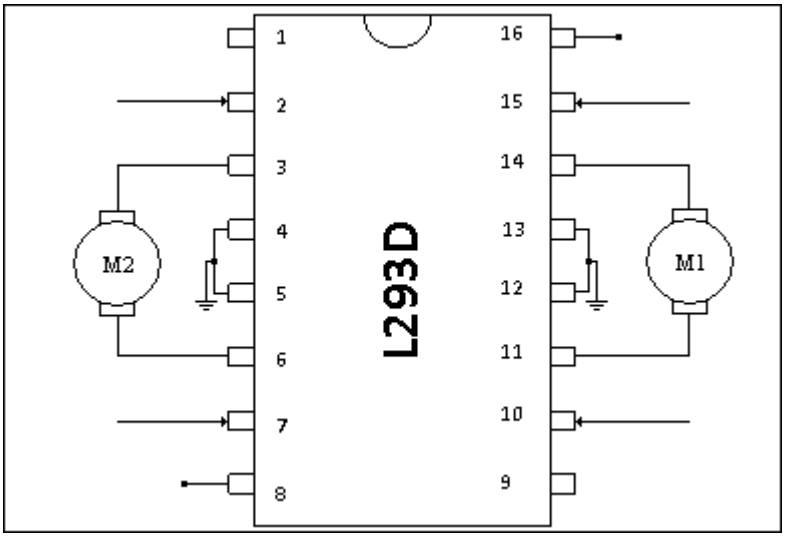


Figura 2.7 *Circuito interno L293D*

Cada uno de los pines que se representan en la figura 2.9 corresponden a:

1. Control de la tensión que se le entrega al motor. Conectada a una señal PWM del Arduino
2. Control de giro del motor 2
3. Conexión directa al motor 2
4. Conexión a tierra
5. Conexión a tierra
6. Conexión directa al motor 2
7. Control de giro del motor 2
8. Entrada de la tensión de alimentación de los motores
9. Control de la tensión que se le entrega al motor. Conectada a una señal PWM del Arduino
10. Control de giro del motor 1
11. Conexión directa al motor 1
12. Conexión a tierra
13. Conexión a tierra
14. Conexión directa al motor 1
15. Control de giro del motor 1
16. Alimentación del integrado

2.2.4 Servomotores MS-1.3-R

Para poder conseguir el movimiento del móvil, es necesario hacer de uso de estos servomotores de rotación continua, los cuales son activados convenientemente gracias al puente H del L293D y permiten el desplazamiento tanto en línea recta, como marcha atrás como desplazamientos hacia los lados.

Cada uno de estos motores va acoplado al eje de unas llantas, las cuales girarán solidarias al movimiento del eje del servo

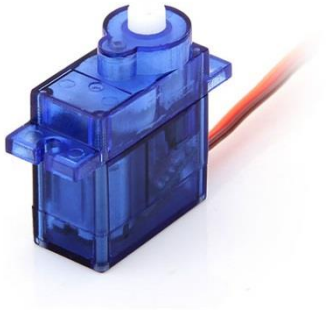


Figura 2.8 *Servomotor MS-1.3-R*

2.2.5 Protoboard

El montaje del arduino así como de todos los elementos electrónicos se realizará sobre una protoboard o placa de pruebas. Para la realización de este proyecto se ha optado por una protoboard con un total de 830 pines, como la que ve en la imagen.

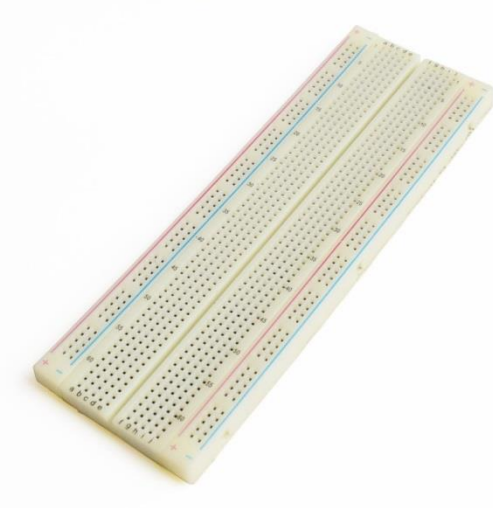


Figura 2.9 *Protoboard con 830 pines*

2.2.6 Conjunto chasis y ruedas

Una vez realizado todo el montaje electrónico, será necesario armarlo a un chasis que actuará como cuerpo de nuestro vehículo. Hemos optado por un modelo distribuido por OLIMEX LTD, el cual incluye dos ruedas para el eje trasero así como una rueda libre para la parte frontal.

Figura 2.10 *Chasis Aluminio*

Figura 2.11 *Rueda loca*

2.2.7 Pila 9V

Por último, se hace uso de una pila de 9V junto a un adaptador para poder alimentar tanto al Arduino, al módulo bluetooth como a los motores de corriente continua que se encargan de generar el movimiento de las ruedas traseras del vehículo.

Figura 2.12 *Pila 9V*

Figura 2.13 *Conector pila 9V*

2.3 Descripción del Software utilizado

Para el desarrollo del proyecto se ha utilizado el software de Arduino y el lenguaje de programación WIRING.

2.3.1 Lenguaje WIRING

WIRING es un lenguaje de programación basado en un entorno de JAVA desarrollado por el denominado Arduino Software (IDE), el cual permite realizar un código sencillo y de fácil entendimiento.

Un programa en WIRING se puede dividir en tres partes principales: estructura, valores (variables y constantes) y funciones.

En la estructura declaramos si queremos que la función sea mediante **setup()**, de forma que solo se ejecute una vez, o un **loop()**, con lo cual se ejecutará de forma indefinida, hasta que lo detengamos de forma manual.

En la estructura también es donde se hace uso de los condicionales, punteros, etc.

Las variables, al igual que en otros tipos de lenguajes de programación, pueden ser de varios tipos, destacando principalmente **void**, **char** e **int**.

Las funciones son las responsables de que se ejecute el código, y por lo tanto las encargadas de que se realicen las diferentes acciones que queremos que se den en nuestro microcontrolador.

A continuación se presenta un ejemplo de código que se puede encontrar en la web de Arduino y el cual realiza el encendido y apagado del LED que lleva incorporado el propio controlador

```
/*
 Turns on an LED for one second, then off for one second, repeatedly.

 This example code is in the public domain.
*/

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;    // set the variable "led" to the value 13

// the setup routine runs once when you press reset:
void setup() {
    // initialize the digital pin as an output.
    pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);  // turn the LED on
    delay(1000);              // wait for a second
    digitalWrite(led, LOW);   // turn the LED off
    delay(1000);              // wait for a second
}
```

Figura 2.14 *Estructura de una tarea en WIRING.*

Para transmitir nuestro código al controlador, basta con descargarse del programa disponible en la web de Arduino, y conectando mediante USB nuestro controlador a un PC con dicho programa bastará con utilizar la opción **Subir** que se encuentra junto a la tecla para verificar el código.

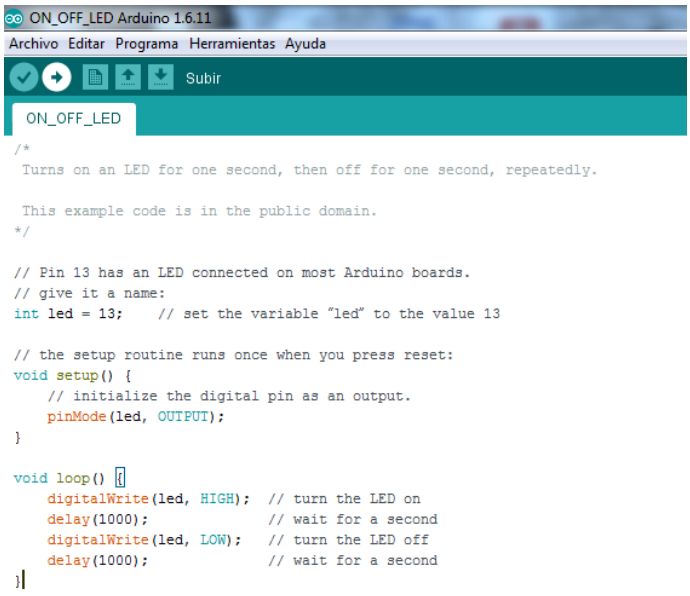


Figura 2.15 *Cargando nuestro código al controlador.*

2.3.2 Software Arduino

Arduino es el software utilizado por la empresa de mismo nombre para crear todo el código que luego le será transmitido a la placa para así poder llevar a cabo las operaciones deseadas.

De interfaz muy sencilla, apta tanto para gente experta en programación como para gente recién iniciada en el mundo, permite generar un código de manera fácil y rápida.

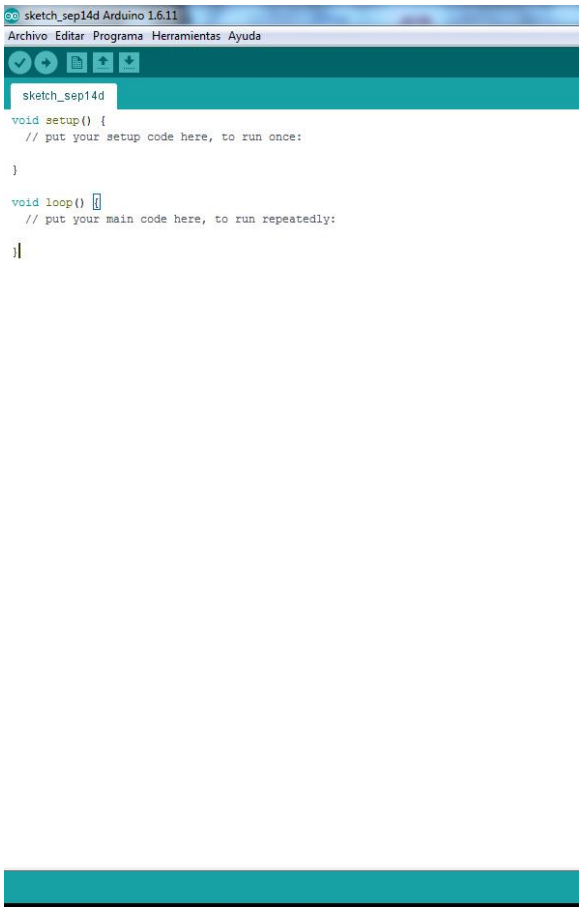


Figura 2.16 *Interfaz Arduino.*

Las diferentes pestañas que dispone la interfaz gráfica de usuario del Arduino, desde las cuales nos desplazamos a todos los ámbitos de trabajo, es la que muestra la siguiente imagen:

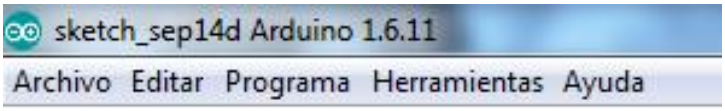


Figura 2.17 *Pestañas Interfaz Arduino.*

ARCHIVO: Contiene las opciones necesarias para crear un nuevo proyecto, guardar el proyecto o imprimirlo.

EDITAR: Contiene las opciones para poder editar la línea de texto del código, así como buscar líneas exactas del código.

PROGRAMA: Permite realizar la compilación del programa, así como subirlo y cargarlo a nuestro controlador.

HERRAMIENTAS: Permite elegir el modelo de controlador con el cual queremos realizar la simulación, así como la opción de reparar el código.

AYUDA: En esta pestaña encontramos información sobre el entorno, posibles errores y sobre la empresa Arduino así como un link a su página oficial.

3 Planteamiento del problema a resolver

3.1 Introducción

En el siguiente capítulo se expone uno de los problemas más comunes en el ambiente de la ingeniería y porqué es necesario el uso de estos móviles

Como ya se ha mencionado en el primer capítulo, uno de los objetivos que busca la ingeniería de sistemas y automática es conseguir proyectos en los cuales la intervención del ser humano sea inexistente o lo menor posible.

Uno de los problemas que se pretende solucionar en este proyecto puede ser el que se tenga una avería en un lugar inaccesible para una persona física, pero que gracias al móvil se pueda acceder a ella y realizar una estimación de la gravedad del fallo para así poder tomar las medidas que se crean oportunas.

3.2 Descripción de incidencias

La solución al caso expuesto anteriormente pasa por realizar el diseño y la construcción de un elemento motorizado que sea capaz de llegar a dichas zonas de difícil acceso para el hombre.

Los casos en los que se pueda dar este problema pueden ser:

- Posible fuga de gas/agua en una zona de difícil acceso.
- Avería de algún tipo de maquinaria, ya sea por pérdida de aceite o de combustible.

Para prevenir los accidentes es necesario saber ante qué tipo de percances nos encontramos, para así poder analizar cómo actuar ante él y evitar riesgos innecesarios.

4 Análisis de las Alternativas

4.1 Introducción

Una vez definido el problema es necesario encontrar la solución más óptima entre todas las posibles, e implementarla.

En este capítulo se hace una recolección de todas las alternativas posibles, detallando cada una de ellas para, por último, hacer una comparativa entre ellas y elegir la mejor opción.

4.2 Posibles Opciones de Mejora

Existen numerosas formas de realizar el diseño del móvil autoguiado, y cada una sería tan válida como la anterior. Dicho esto, una de las múltiples mejoras que se le podrían incluir serían:

- Inclusión de un módulo ultrasonido para conseguir que el móvil registre si tiene obstáculos en su trayectoria y así estudiar una ruta alternativa para esquivarlo.
- Dotación de una pequeña cámara que suministre información tanto en sonido como en imagen del entorno en el cual se está moviendo el móvil.

4.3 Descripción Detallada de Opciones de Mejora

A continuación se explica con detalle las alternativas que se han mencionado más arriba, para que se pueda hacer un análisis a posteriori de cuál es la más óptima.

La primera opción trata de incorporar un sensor ultrasonido en la parte frontal del chasis del vehículo para así poder saber si hay algún tipo de obstáculo y, si en caso de poder ser evitado, estudiar una trayectoria distinta la cual permita salvar el obstáculo.

Para su correcto funcionamiento es necesario suministrarle 5V, así como un pulso de 10 µs a través del pin TRIG para así poder activarlo.

Una vez hecho esto, el módulo lanzará una ráfaga de 8 pulsos a una frecuencia de 40Khz y la salida ECHO pasará a un nivel alto hasta que el módulo reciba un eco, momento en el cual pasará al nivel bajo. Es decir, la salida ECHO será un pulso cuyo ancho es proporcional a la distancia respecto a un objeto. Si no se detecta un objeto, la salida ECHO pasara a nivel bajo después de 30ms.

Si el ancho del pulso se mide en µs, el resultado se debe dividir entre 58 para saber la distancia en centímetros, y entre 148 para saber la distancia en pulgadas. Estos valores son obtenidos de:

Si la velocidad del sonido es 340 metros por segundo o 29µs por centímetro, y como el sonido tiene que viajar dos veces la distancia hacia el objeto, una de ida y otra de vuelta, entonces cada 2x29=58µs recorrerá un centímetro.

El modulo debe activarse cada 50ms como mínimo, de esta manera se asegura que la ráfaga ultrasónica haya desaparecido completamente y no provocará un falso eco en la siguiente medición de distancia.

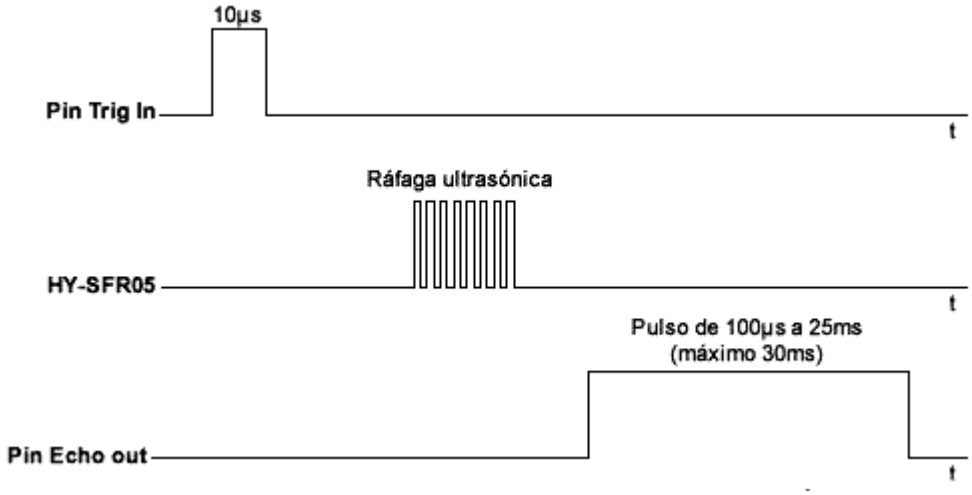


Figura 4.1 *Espectro de ondas del módulo ultrasónico.*

El código a implementar sería algo similar a éste:

```
#include <NewPing.h>

/*Aqui se configuran los pines donde debemos conectar el sensor*/
#define TRIGGER_PIN  12
#define ECHO_PIN     11
#define MAX_DISTANCE 200

/*Crear el objeto de la clase NewPing*/
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
  Serial.begin(9600);
}

void loop() {
  // Esperar 1 segundo entre mediciones
  delay(1000);
  // Obtener medicion de tiempo de viaje del sonido y guardar en variable uS
  int uS = sonar.ping_median();
  // Imprimir la distancia medida a la consola serial
  Serial.print("Distancia: ");
  // Calcular la distancia con base en una constante
  Serial.print(uS / US_ROUNDTRIP_CM);
  Serial.println("cm");
}
```

Figura 4.2 *Código Módulo Ultrasónico .*

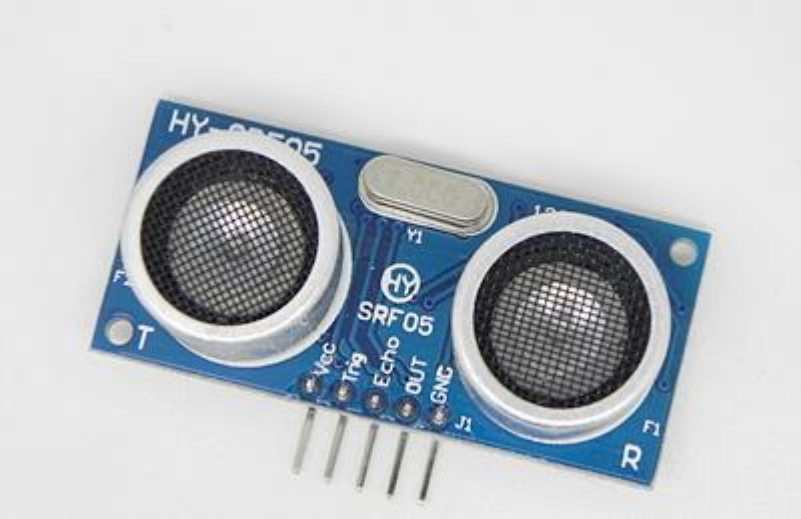


Figura 4.3 *Módulo Ultrasonidos Arduino .*

La otra opción que se puede estudiar es la incorporación de un dispositivo de grabación de imagen al procesador de Arduino, para así poder seguir realizando las mismas funciones con el vehículo pero con la posibilidad de poder tener imagen a tiempo real del itinerario que está siguiendo el móvil.

Para este caso bastaría con hacer las conexiones pertinentes para que haya tensión en la cámara y así poder grabar imagen, amén de un sistema de almacenamiento para poder almacenar todo lo que se filme en el proceso (una tarjeta microSD debidamente conectada con su módulo sería suficiente).

Figura 4.4 *Módulo Cam de Arduino .*

4.4 Priorización de Opciones

Una vez ya detalladas todas las alternativas, es necesario hacer una comparativa para elegir cuál va a ser la mejor opción que se lleve a cabo.

La primera opción es relativamente sencilla de implementar, tiene un coste bajo y ayudaría a evitar problemas que se den in situ mientras se está manipulando el vehículo.

La segunda opción también es de implementación sencilla, pero habría que añadirles los costes que suponen la cámara (más elevado que el del módulo ultrasonidos), así como el del dispositivo de almacenamiento, como por ejemplo un módulo que soporte tarjetas microSD.

Siendo ambas opciones perfectamente compatibles, lo más viable tanto en términos de montaje como económicos sería optar por la primera opción e incluir un sensor de ultrasonidos al vehículo.

5 Descripción de la Solución

5.1 Introducción

El siguiente paso después de haber elegido la solución al problema más óptima es realizar el diseño y el montaje del móvil autoguiado.

5.2 Diseño y Desarrollo de la Aplicación

En este capítulo se explica cuál ha sido el diseño y el desarrollode la cosntrucción del móvil, por un lado la configuración del controlador y por otro la programación en WIRING del CPU. Para, por último, realizar las pruebas necesarias y comprobar cuáles son los resultados del ensayo.

5.2.1 Configuración del Controlador

En primer lugar debemos crear un esquema en el cual plasmemos todas las conexiones que hay que realizar para garantizar el funcionamiento con éxito tanto del Arduino, como del módulo bluetooth como del L293D que se encargará de mandar las órdenes a los motores para que realicen los giros correspondientes.

Para ello indicaremos planos simplificados de las conexiones a continuación, con los planos en su debido formato en el anexo que se proporciona al final.

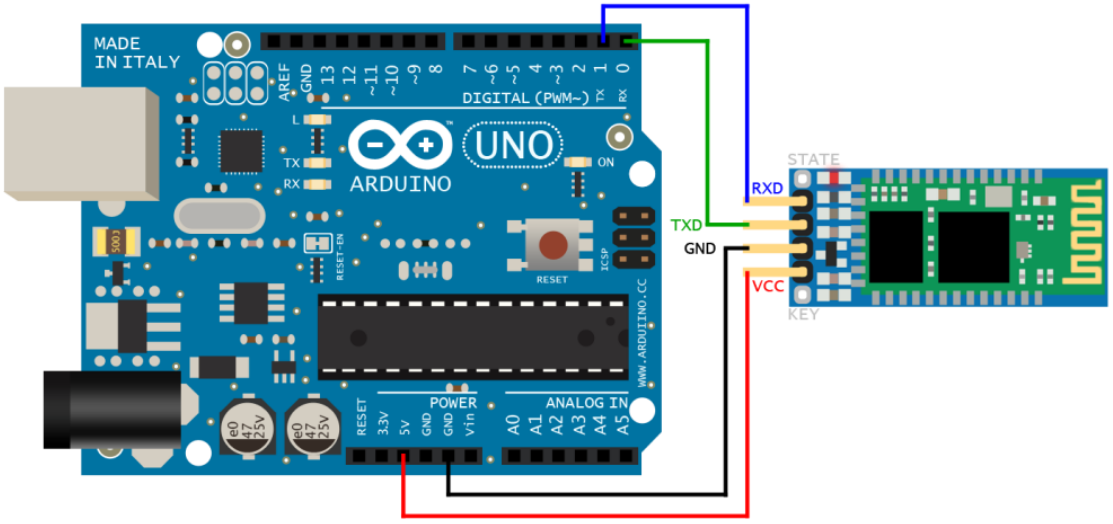


Figura 5.1 *Conexión Arduino con HC-06.*

Como se puede observar, para obtener comunicación entre el arduino y el módulo bluetooth basta con conectar los terminales de tierra (GND), el Vcc del HC-06 a la entrada de 5V del arduino, y el Tx del HC-06 con el Rx del arduino y el Tx del arduino con el Rx del HC-06.

Ahora hay que conectar el circuito integrado L293D con el Arduino y con los motores, para así poder tener total control sobre los mismos.

Anteriormente se dijo para qué servían cada una de las patas del L293D, con lo que simplemente basta con realizar las conexiones pertinentes, quedando el siguiente resultado.

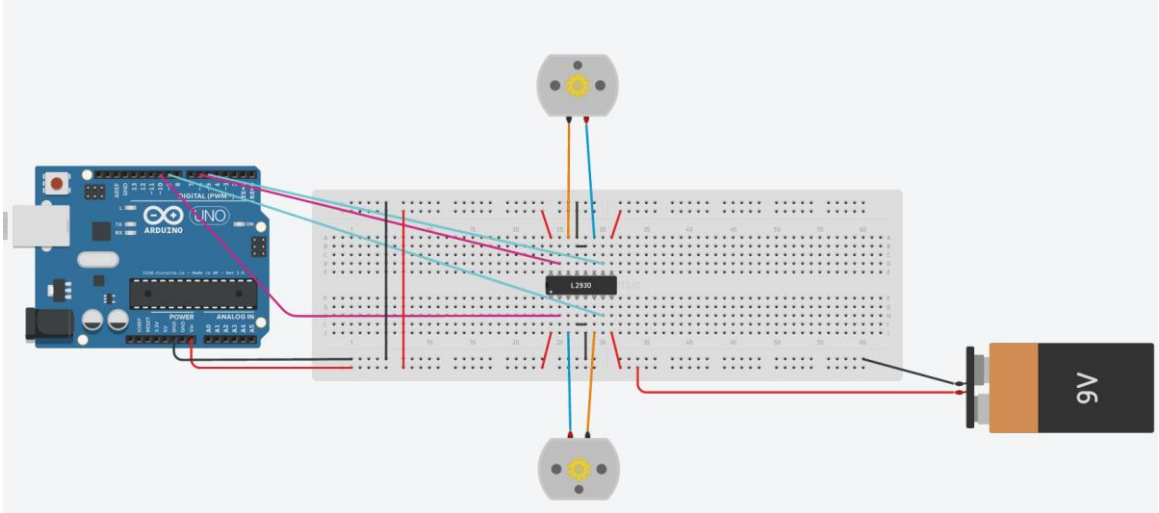


Figura 5.2 *Configuración puente H con motores y CPU.*

Una vez realizado esto, el siguiente paso es implementar el código para hacer funcionar el montaje.

Pero antes de ello, para comprobar el correcto funcionamiento del puente H, se genera un código el cual genera la siguiente secuencia de comandos:

- Tras un delay de 0,5 segundos, ambos motores se ponen en marcha hacia delante durante una duración de 2 segundos.
- Transcurrido este tiempo, ambos motores se detienen, y actúa sólo uno para simular un giro hacia la derecha durante 0,5 segundos.
- Tras parar el motor, esta vez girará el otro motor para simular un giro hacia la izquierda durante 0,5 segundos.
- Una vez detenido el motor, ambos motores se ponen en marcha pero en sentido contrario al del primer caso, es decir, irán marcha atrás durante 2 segundos.

```
int izqA = 5;
int izqB = 6;
int derA = 9;
int derB = 10;
int vel = 5; // Velocidad de los motores (0-255)

void setup()  {
  pinMode(derA, OUTPUT);
  pinMode(derB, OUTPUT);
  pinMode(izqA, OUTPUT);
  pinMode(izqB, OUTPUT);
 }

void loop()  {
    analogWrite(derB, 0);
    analogWrite(izqB, 0);
    delay (500);
    analogWrite(derA, vel);
    analogWrite(izqA, vel);
    delay (2000);

    analogWrite(derA, vel);
    analogWrite(izqA, 0);
    delay (500);

    analogWrite(derA, 0);
    analogWrite(izqA, vel);
    delay (500);

    analogWrite(derA, 0);
    analogWrite(izqA, 0);
    delay (500);
    analogWrite(derB, vel);
```

Figura 5.3 *Código para comprobación funcionamiento puente H.*

5.2.2 Creación del Código WIRING

La segunda parte del desarrollo, una vez realizadas las conexiones físicas, consiste en crear el código para hacer funcionar nuestro conjunto Arduino más los demás componentes electrónicos.

Para poder empezar a realizar el código, deberemos primero asignar los pines con posibilidad de PWM que dispone el Arduino para poder utilizarlos. Estos pines son el 3, 5, 6, 10 y 11. Para realizar un correcto uso de nuestro móvil bluetooth bastará con elegir cuatro de estos pines.

Una vez hecho esto, asignamos un valor de variable **int** así como un nombre a nuestro gusto para cada uno de los pines PWM que vamos a utilizar (por ejemplo **int** izqA = 5 ; para así denominar al pin 5 izqA).

También asignamos la velocidad a la que queremos que giren los motores (en un rango entre 0 y 255).

A continuación procedemos a iniciar el puerto serial para poder establecer comunicación entre el CPU y el módulo bluetooth. Para ello haremos uso de una función **setup()** en la cual también indicaremos que los pines del PWM sean tratados como Outputs con el comando **pinMode(pin,mode)**.

Procedemos a iniciar la estructura principal de nuestro código, mediante el cual generamos los comandos necesarios para poder realizar el control del móvil desde un teléfono que posea la aplicación necesaria así como bluetooth.

Para ello se genera un bucle **loop()** el cual posee otros bucles condicionales con la estructura **if()** que realizan las siguientes acciones:

- Leer el bluetooth y almacenar en la variable **estado**
- Si la variable estado tiene el valor **a** (asignado al botón de frente de nuestra app), ambos motores giraran en el mismo sentido y provocarán el movimiento hacia delante del móvil.
- Si la variable estado tiene el valor **b** (asignado al botón izquierdo de nuestra app), solo girará el motor de la izquierda, lo cual provocará que el móvil se desplace hacia la derecha.
- Si la variable estado tiene el valor **c** (asignado al botón stop de nuestra app), ambos motores se detendrán y el móvil permanecerá en reposo.
- Si la variable estado tiene el valor **d** (asignado al botón derecha de nuestra app), solo girará el motor de la derecha, lo cual provocará que el móvil se desplace hacia la izquierda.
- Si la variable estado tiene el valor **e** (asignado al botón de marcha atrás de nuestra app), ambos motores giraran en el mismo sentido y provocarán el movimiento hacia atrás del móvil.

```
void loop()  {

  if(Serial.available()>0){
      estado = Serial.read();
  }
  if(estado=='a'){
      analogWrite(derB, 0);
      analogWrite(izqB, 0);
      analogWrite(derA, vel);
      analogWrite(izqA, vel);
  }
  if(estado=='b'){
      analogWrite(derB, 0);
      analogWrite(izqB, 0);
      analogWrite(derA, 0);
      analogWrite(izqA, vel);
  }
  if(estado=='c'){
      analogWrite(derB, 0);
      analogWrite(izqB, 0);
      analogWrite(derA, 0);
      analogWrite(izqA, 0);
  }
  if(estado=='d'){
       analogWrite(derB, 0);
       analogWrite(izqB, 0);
       analogWrite(izqA, 0);
       analogWrite(derA, vel);
  }

  if(estado=='e'){
       analogWrite(derA, 0);
       analogWrite(izqA, 0);
       analogWrite(derB, vel);
       analogWrite(izqB, vel);
  }
}
```

Figura 5.4 *Estructura bucle principal.*

Una vez realizado esto, basta con cargar el sketch (nombre que utiliza Arduino para el programa) a nuestro CPU a través del puerto USB y del software que podemos encontrar en la web de Arduino. Pero es **IMPRESCINDIBLE** que a la hora de cargar el sketch el módulo bluetooth no esté conectado con la CPU, pues nos puede generar errores y un incorrecto funcionamiento.

5.2.3 Pruebas y Resultados

Por último, después de toda la fase de investigación y desarrollo de la aplicación, es necesaria la fase de pruebas y resultados para comprobar que todo funcione correctamente.

Esta fase es recomendable realizarla nada más se ha terminado el montaje del proyecto para así poder comprobar si existe algún tipo de fallo (ya sea mecánico o problema con la implementación del código) y corregirlo lo más rápido posible.

Una vez se ha comprobado de manera satisfactoria que no existen errores de ningún tipo, se puede proceder a llevar el móvil al destinatario final donde vaya a ser utilizado.

En las siguientes imágenes se puede observar el resultado final del vehículo una vez realizado el montaje electrónico y ajustado al chasis de aluminio.

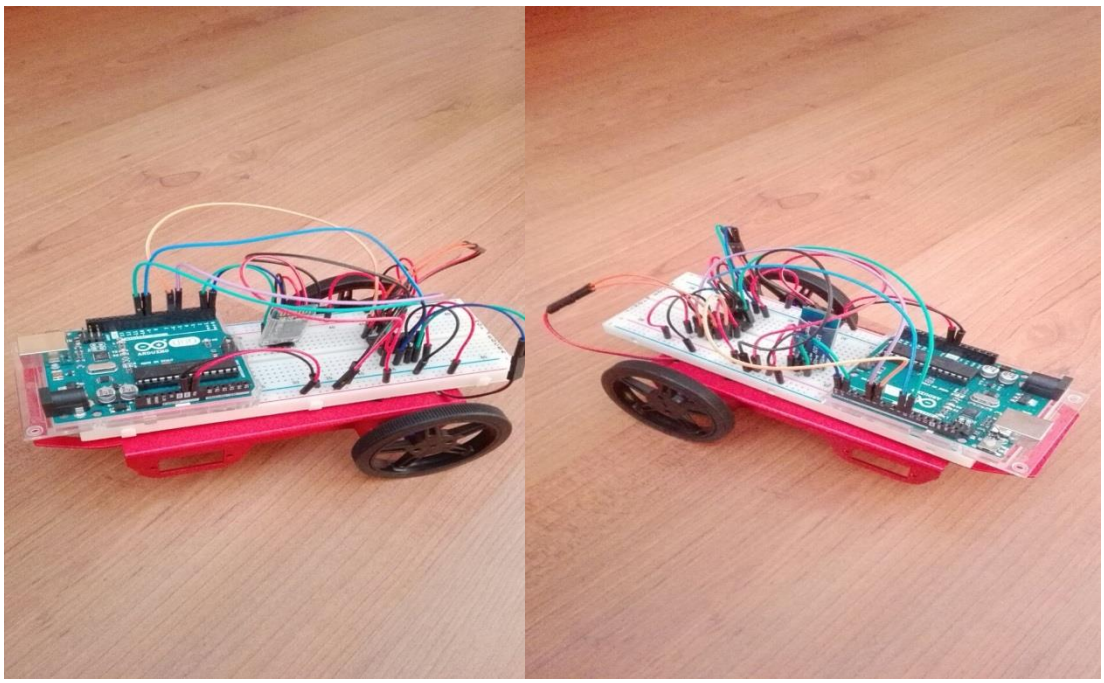


Figura 5.3 *Vista en detalle del móvil una vez montado.*

6 Conclusiones

6.1 Conclusiones

Tras diseñar e implementar el móvil, permitiendo realizar un control total del mismo gracias a un dispositivo bluetooth y una aplicación sin necesidad de estar junto a él, se ha verificado el correcto funcionamiento de dicha aplicación con el móvil, quedando satisfechos los objetivos propuestos.

Ha sido un trabajo paciente de investigación y desarrollo, principalmente sobre toda la parte de implementación del código puesto que se ha hecho uso de todas las ayudas disponibles en la página web de Arduino, y por último una fase de pruebas ante distintos errores que iban surgiendo conforme se avanzaba en el proyecto.

6.2 Líneas Futuras de Mejora

Aun habiendo cumplido satisfactoriamente con todos los objetivos marcados para este proyecto, siguen habiendo aspectos mejorables en el diseño, y se proponen las siguientes líneas futuras de trabajo para aquellos interesados en la automatización:

- Implementación de un sensor de ultrasonidos que ayude a evitar obstáculos de manera autónoma.
- Implementación de un dispositivo con captación de video para poder tener información a tiempo real de la trayectoria del móvil.
- Diseño de un chasis más ergonómico y “tradicional” para el móvil, en el cual se usen 4 ruedas lo cual permitiría un control más estable del mismo.

7 Referencias Bibliográficas

- ***Libros***

[1] **Introducción a Arduino**, Massimo Banzi; ANAYA.

- ***Manuales***

[2] **MANUAL DE PROGRAMACIÓN EN ARDUINO.** Jose Manuel Ruiz Gutiérrez

- ***Webs***

[3] **ARDUINO GUIDES**

https://www.arduino.cc/en/Reference/HomePage

[4] **AUTODESK CIRCUITS**

https://circuits.io/lab

[5] **OLIMEX LTD**

https://www.olimex.com/

[6] Cómo estructurar y escribir un TFG. **Jose P. García Sabater.**

http://jpgarcia.webs.upv.es/?page_id=34

8 Anexos

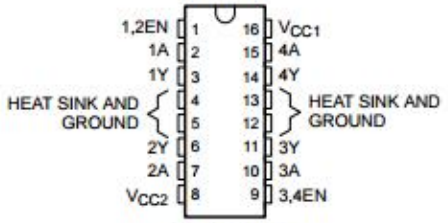
8.1 DataSheet L293D.

L293, L293D
QUADRUPLE HALF-H DRIVERS

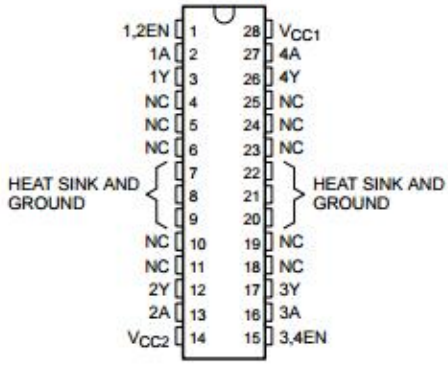
SLRS008B – SEPTEMBER 1986 – REVISED JUNE 2002

- **Featuring Unitrode L293 and L293D Products Now From Texas Instruments**
- **Wide Supply-Voltage Range: 4.5 V to 36 V**
- **Separate Input-Logic Supply**
- **Internal ESD Protection**
- **Thermal Shutdown**
- **High-Noise-Immunity Inputs**
- **Functional Replacements for SGS L293 and SGS L293D**
- **Output Current 1 A Per Channel (600 mA for L293D)**
- **Peak Output Current 2 A Per Channel (1.2 A for L293D)**
- **Output Clamp Diodes for Inductive Transient Suppression (L293D)**

N, NE PACKAGE
(TOP VIEW)



DWP PACKAGE
(TOP VIEW)



description

The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

On the L293, external high-speed output clamp diodes should be used for inductive transient suppression.

A V_{CC1} terminal, separate from V_{CC2}, is provided for the logic inputs to minimize device power dissipation.

The L293and L293D are characterized for operation from 0°C to 70°C.

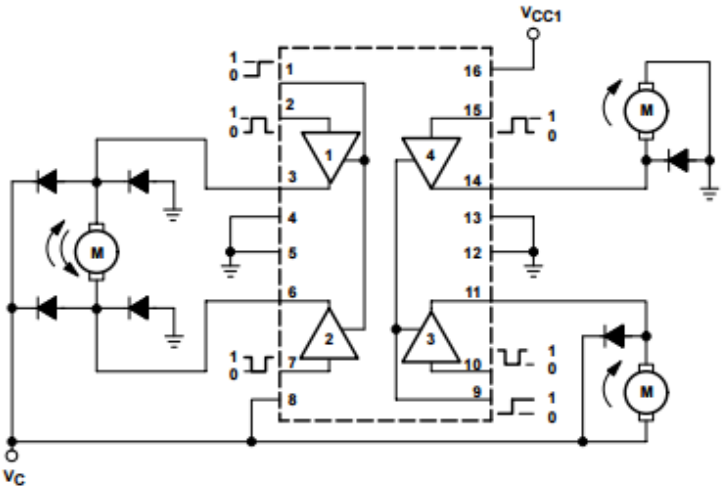
TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

1

L293, L293D
QUADRUPLE HALF-H DRIVERS

SLRS008B – SEPTEMBER 1986 – REVISED JUNE 2002

block diagram



NOTE: Output diodes are internal in L293D.

TEXAS INSTRUMENTS AVAILABLE OPTIONS

T_A	PACKAGE
	PLASTIC DIP (NE)
0°C to 70°C	L293NE L293DNE

Unitrode Products from Texas Instruments
AVAILABLE OPTIONS

T_A	PACKAGED DEVICES	
	SMALL OUTLINE (DWP)	PLASTIC DIP (N)
0°C to 70°C	L293DWP L293DDWP	L293N L293DN

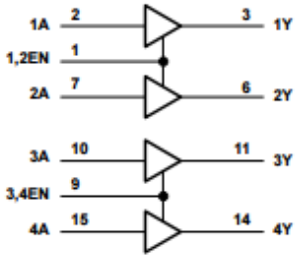
The DWP package is available taped and reeled. Add the suffix TR to device type (e.g., L293DWPTR).

FUNCTION TABLE
(each driver)

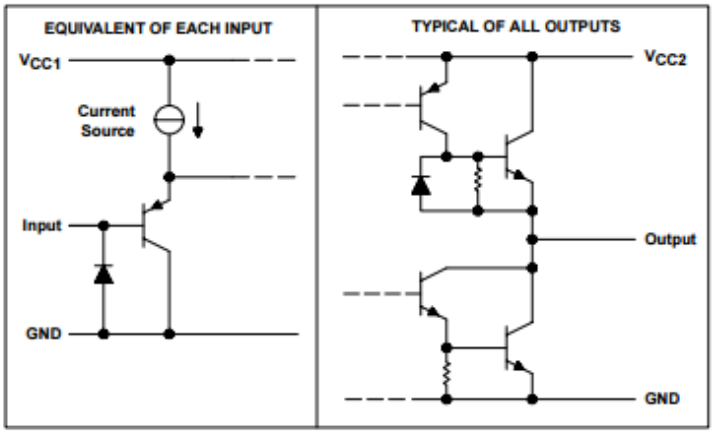
INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = high level, L = low level, X = irrelevant, Z = high impedance (off)
† In the thermal shutdown mode, the output is in the high-impedance state, regardless of the input levels.

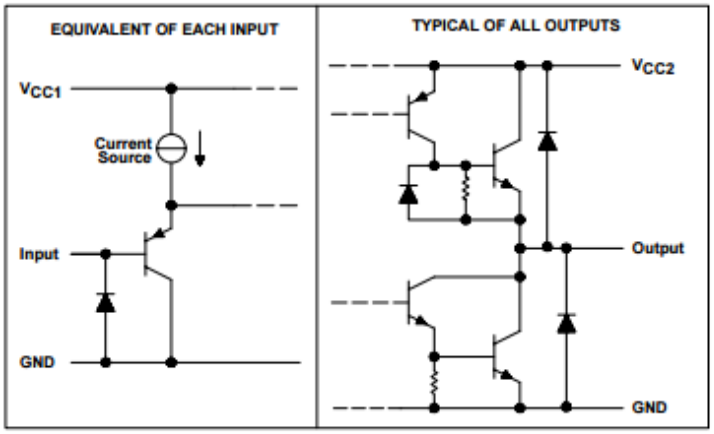
logic diagram



schematics of inputs and outputs (L293)



schematics of inputs and outputs (L293D)



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{CC1} (see Note 1)	36 V
Output supply voltage, V_{CC2}	36 V
Input voltage, V_I	7 V
Output voltage range, V_O	–3 V to V_{CC2} + 3 V
Peak output current, I_O (nonrepetitive, t ≤ 5 ms): L293	±2 A
Peak output current, I_O (nonrepetitive, t ≤ 100 µs): L293D	±1.2 A
Continuous output current, I_O: L293	±1 A
Continuous output current, I_O: L293D	±600 mA
Continuous total dissipation at (or below) 25°C free-air temperature (see Notes 2 and 3)	2075 mW
Continuous total dissipation at 80°C case temperature (see Note 3)	5000 mW
Maximum junction temperature, T_J	150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds	260°C
Storage temperature range, T_{stg}	–65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES: 1. All voltage values are with respect to the network ground terminal.
2. For operation above 25°C free-air temperature, derate linearly at the rate of 16.6 mW/°C.
3. For operation above 25°C case temperature, derate linearly at the rate of 71.4 mW/°C. Due to variations in individual device electrical characteristics and thermal resistance, the built-in thermal overload protection may be activated at power levels slightly above or below the rated dissipation.

recommended operating conditions

			MIN	MAX	UNIT
	Supply voltage	V_{CC1}	4.5	7	V
		V_{CC2}	V_{CC1}	36	
V_{IH}	High-level input voltage	$V_{CC1} \le 7$ V	2.3	V_{CC1}	V
		$V_{CC1} \ge 7$ V	2.3	7	V
V_{IL}	Low-level output voltage		−0.3†	1.5	V
T_A	Operating free-air temperature		0	70	°C

† The algebraic convention, in which the least positive (most negative) designated minimum, is used in this data sheet for logic voltage levels.

electrical characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER			TEST CONDITIONS		MIN	TYP	MAX	UNIT
V_{OH}	High-level output voltage		L293: I_{OH} = −1 A L293D: I_{OH} = −0.6 A		V_{CC2}−1.8	V_{CC2}−1.4		V
V_{OL}	Low-level output voltage		L293: I_{OL} = 1 A L293D: I_{OL} = 0.6 A			1.2	1.8	V
V_{OKH}	High-level output clamp voltage		L293D: I_{OK} = −0.6 A			V_{CC2} + 1.3		V
V_{OKL}	Low-level output clamp voltage		L293D: I_{OK} = 0.6 A			1.3		V
I_{IH}	High-level input current	A	V_I = 7 V			0.2	100	µA
		EN				0.2	10	
I_{IL}	Low-level input current	A	V_I = 0			−3	−10	µA
		EN				−2	−100	
I_{CC1}	Logic supply current		I_O = 0	All outputs at high level		13	22	mA
				All outputs at low level		35	60	
				All outputs at high impedance		8	24	
I_{CC2}	Output supply current		I_O = 0	All outputs at high level		14	24	mA
				All outputs at low level		2	6	
				All outputs at high impedance		2	4	

switching characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER		TEST CONDITIONS	L293NE, L293DNE			UNIT
			MIN	TYP	MAX	
t_{PLH}	Propagation delay time, low-to-high-level output from A input	C_L = 30 pF, See Figure 1		800		ns
t_{PHL}	Propagation delay time, high-to-low-level output from A input			400		ns
t_{TLH}	Transition time, low-to-high-level output			300		ns
t_{THL}	Transition time, high-to-low-level output			300		ns

switching characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER		TEST CONDITIONS	L293DWP, L293N L293DDWP, L293DN			UNIT
			MIN	TYP	MAX	
t_{PLH}	Propagation delay time, low-to-high-level output from A input	C_L = 30 pF, See Figure 1		750		ns
t_{PHL}	Propagation delay time, high-to-low-level output from A input			200		ns
t_{TLH}	Transition time, low-to-high-level output			100		ns
t_{THL}	Transition time, high-to-low-level output			350		ns

8.2 DataSheet Arduino UNO.

Technical specs

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

8.3 DataSheet HC-06.

Guangzhou HC Information Technology Co., Ltd.

1. Product's picture

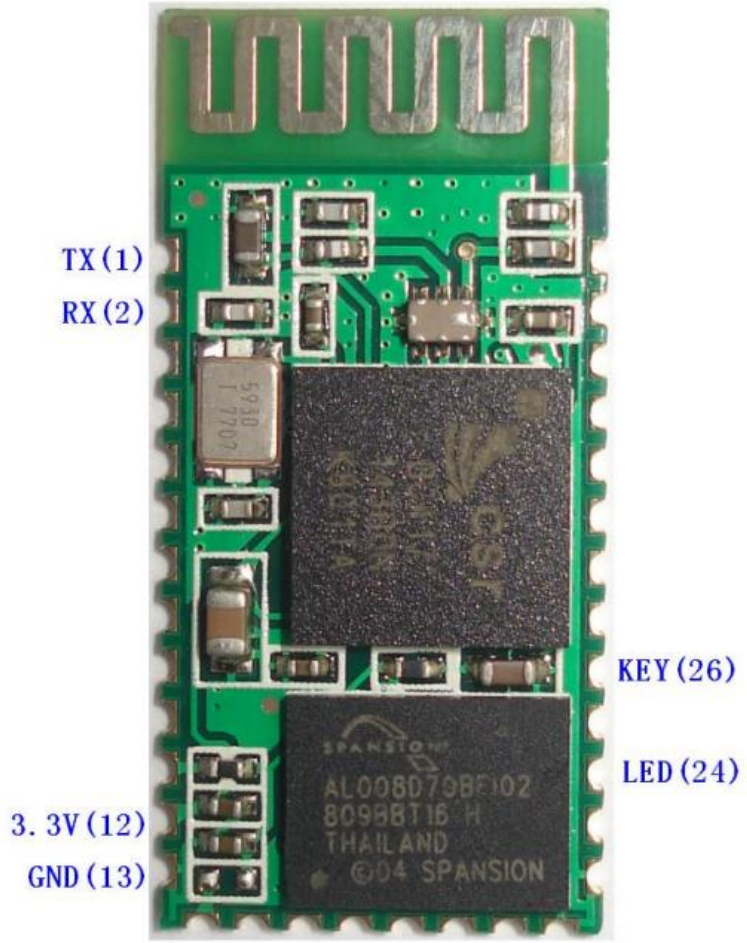


Figure 1 A Bluetooth module

Guangzhou HC Information Technology Co., Ltd.

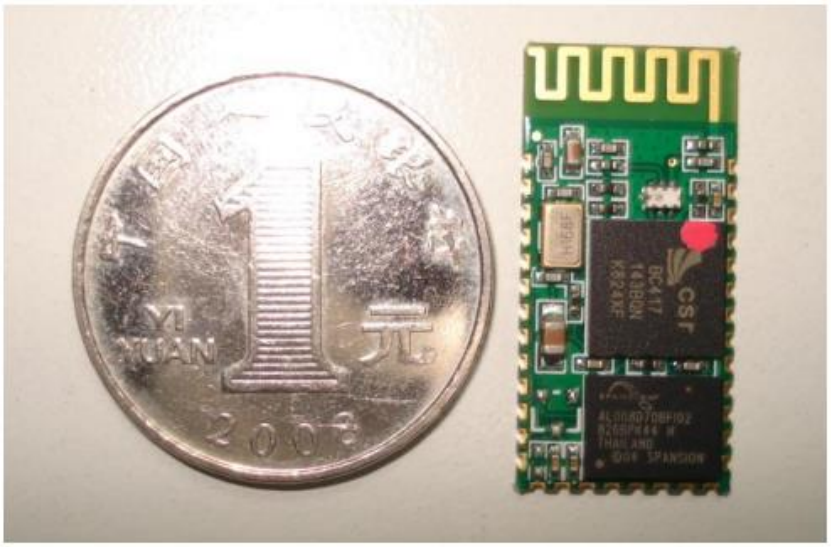


Figure 2. A Bluetooth module size

Figure 3 50 pieces chips in an anti-static blister package.

2. Feature

- Wireless transceiver
  - Sensitivity (Bit error rate) can reach -80dBm.
  - The change range of output's power: -4 - +6dBm.
- Function description (perfect Bluetooth solution)
  - Has an EDR module; and the change range of modulation depth: 2Mbps - 3Mbps.
  - Has a build-in 2.4GHz antenna; user needn't test antenna.
  - Has the external 8Mbit FLASH
  - Can work at the low voltage (3.1V~4.2V). The current in pairing is in the range of 30~40mA. The current in communication is 8mA.
  - Standard HCI Port (UART or USB)
  - USB Protocol: Full Speed USB1.1, Compliant With 2.0
  - This module can be used in the SMD.
  - It's made through RoHS process.
  - The board PIN is half hole size.
  - Has a 2.4GHz digital wireless transceiver.
  - Bases at CSR BC04 Bluetooth technology.
  - Has the function of adaptive frequency hopping.
  - Small (27mm×13mm×2mm)
  - Peripherals circuit is simple.
  - It's at the Bluetooth class 2 power level.
  - Storage temperature range: -40 ℃ - 85℃， work temperature range: -25 ℃ - +75℃
  - Any wave inter Interference: 2.4MHz， the power of emitting: 3 dBm.
  - Bit error rate: 0. Only the signal decays at the transmission link, bit error may be produced. For example, when RS232 or TTL is being processed, some signals may decay.

- Low power consumption
- Has high-performance wireless transceiver system
- Low Cost

- Application fields:
  - Bluetooth Car Handsfree Device
  - Bluetooth GPS
  - Bluetooth PCMCIA , USB Dongle
  - Bluetooth Data Transfer
- Software
  - CSR

3. PINs description

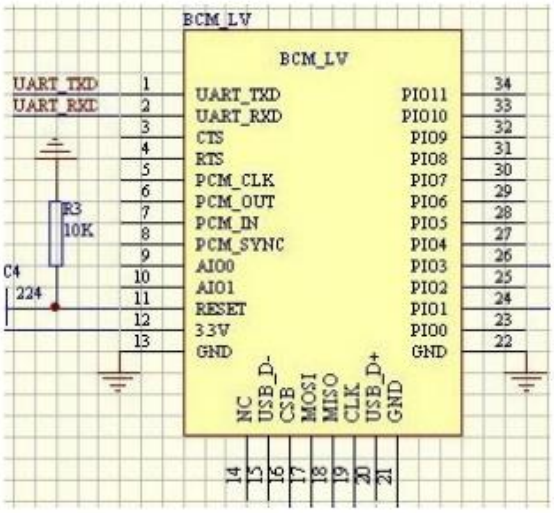


Figure 3 PIN configuration

The PINs at this block diagram is as same as the physical one.

PIN Name	PIN #	Pad type	Description	Note
GND	13 21 22	VSS	Ground pot	
1V8	14	VDD	Integrated 1.8V (+) supply with On-chip linear regulator output within 1.7-1.9V	
VCC	12	3.3V		
AIO0	9	Bi-Directional	Programmable input/output line	
AIO1	10	Bi-Directional	Programmable input/output line	

PIO0	23	Bi-Directional RX EN	Programmable input/output line, control output for LNA(if fitted)	
PIO1	24	Bi-Directional TX EN	Programmable input/output line, control output for PA(if fitted)	
PIO2	25	Bi-Directional	Programmable input/output line	
PIO3	26	Bi-Directional	Programmable input/output line	
PIO4	27	Bi-Directional	Programmable input/output line	
PIO5	28	Bi-Directional	Programmable input/output line	
PIO6	29	Bi-Directional	Programmable input/output line	CLK_REQ
PIO7	30	Bi-Directional	Programmable input/output line	CLK_OUT
PIO8	31	Bi-Directional	Programmable input/output line	
PIO9	32	Bi-Directional	Programmable input/output line	
PIO10	33	Bi-Directional	Programmable input/output line	
PIO11	34	Bi-Directional	Programmable input/output line	
RESETB	11	CMOS Input with weak intemal pull-down		
UART_RTS	4	CMOS output, tri-stable with weak internal pull-up	UART request to send, active low	
UART_CTS	3	CMOS input with weak internal pull-down	UART clear to send, active low	
UART_RX	2	CMOS input with weak internal pull-down	UART Data input	
UART_TX	1	CMOS output, Tri-stable with weak internal pull-up	UART Data output	
SPI_MOSI	17	CMOS input with weak internal pull-down	Serial peripheral interface data input	
SPI_CSB	16	CMOS input with weak internal	Chip select for serial peripheral interface, active low	

		pull-up		
SPI_CLK	19	CMOS input with weak internal pull-down	Serial peripheral interface clock	
SPI_MISO	18	CMOS input with weak internal pull-down	Serial peripheral interface data Output	
USB_-	15	Bi-Directional		
USB_+	20	Bi-Directional		
1.8V	14		1.8V external power supply input	Default : 1.8V internal powe r supply.
PCM_CLK	5	Bi-Directional		
PCM_OUT	6	CMOS output		
PCM_IN	7	CMOS Input		
PCM_SYNC	8	Bi-Directional		

Guangzhou HC Information Technology Co., Ltd.

5. Block diagram

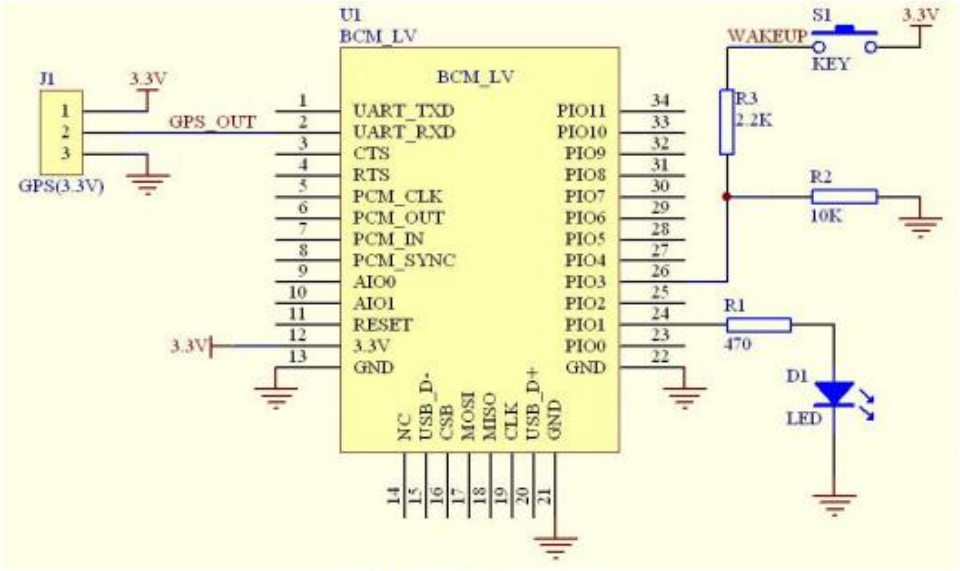


Figure 5 Block diagram 1

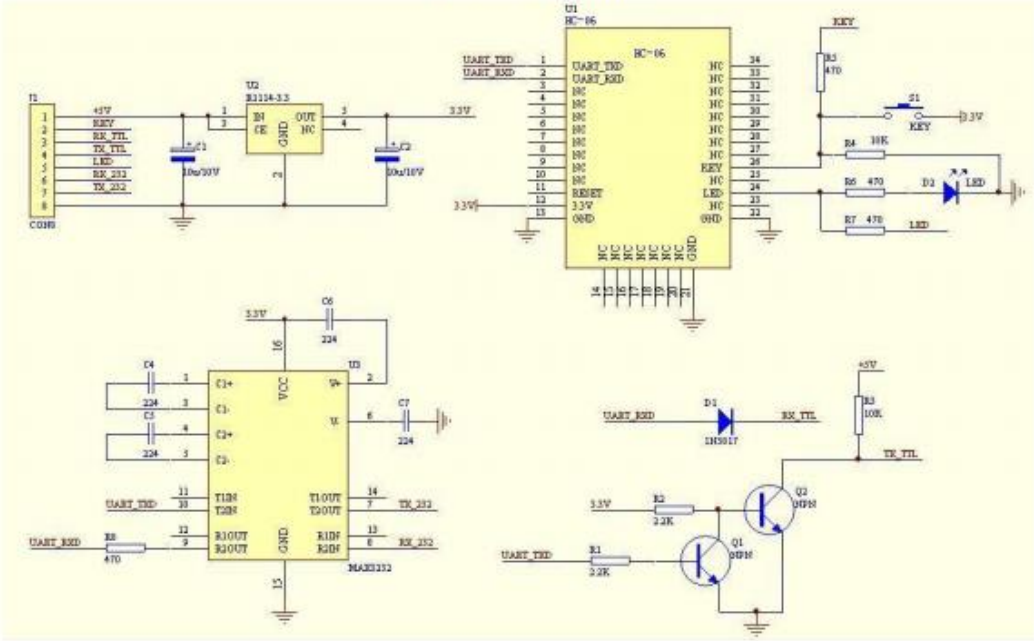


Figure 5 Block diagram 2

8.4 Código WIRING.

```
int izqA = 5;

int izqB = 6;

int derA = 9;

int derB = 10;

int vel = 255;

int estado = 'g';


void setup()  {

 Serial.begin(9600);

 pinMode(derA, OUTPUT);

 pinMode(derB, OUTPUT);

 pinMode(izqA, OUTPUT);

 pinMode(izqB, OUTPUT);

}


void loop()  {


 if(Serial.available()>0){

    estado = Serial.read();

 }
```

```
if(estado=='a'){
    analogWrite(derB, 0);
    analogWrite(izqB, 0);
    analogWrite(derA, vel);
    analogWrite(izqA, vel);
}
if(estado=='b'){
    analogWrite(derB, 0);
    analogWrite(izqB, 0);
    analogWrite(derA, 0);
    analogWrite(izqA, vel);
}
if(estado=='c'){
    analogWrite(derB, 0);
    analogWrite(izqB, 0);
    analogWrite(derA, 0);
    analogWrite(izqA, 0);
}
if(estado=='d'){
     analogWrite(derB, 0);
     analogWrite(izqB, 0);
     analogWrite(izqA, 0);
     analogWrite(derA, vel);
}
```

```
  if(estado=='e'){
      analogWrite(derA, 0);
      analogWrite(izqA, 0);
      analogWrite(derB, vel);
      analogWrite(izqB, vel);
  }

  if  (estado=='g'){
  }
}
```

DOCUMENTO II

PRESUPUESTO

A lo largo de este documento se analiza los costes producidos durante la realización del proyecto. Este estudio económico consta de:

- Coste de Amortización de Equipos.
- Coste de Mano de Obra.
- Coste Total.

1 Coste de Amortización de Equipos

El coste de amortización hace referencia al proceso financiero por medio del cual se reduce una deuda. En el presente proyecto, este tipo de coste aparece en el empleo de los equipos ya instalados que permiten el desarrollo de la aplicación. Con el fin de determinar el nivel de utilización de todos los equipos, se ha aplicado un coeficiente de amortización que responde a la siguiente ecuación:

Coef. Amortización=Tiempo de Proyecto/Tiempo estimado de amortización

En la siguiente tabla aparecen detallados dichos costes.

CONCEPTO	DESCRIPCIÓN	FABRICANTE	PRECIO (€)	COEFICIENTE AMORTIZACIÓN	TOTAL (€)
Hardware	Arduino UNO	ARDUINO	20	0,1	2
Hardware	Ordenador Portátil	ACER	600	0,09	54
Hardware	Módulo HC-06	KEYES	5	0,1	0,5
Software	ARDUINO	ARDUINO	----	----	----
Software	Office Profesional 2010	Microsoft	539	0,07	37,73
Software	Microsoft Windows 10	Microsoft	----	----	----
COSTE TOTAL DE AMORTIZACIÓN DEL EQUIPO					94,23

2 Coste de Mano de Obra

En este apartado se desarrollan los costes derivados de la implementación del proyecto, es decir, todos aquellos costes producidos por el trabajo físico y mental de cada unos de los procesos del mismo. A continuación, se presenta una tabla en la que se detallan dichos costes.

ORGANIZACIÓN DEL PROYECTO			
CONCEPTO	**PRECIO (€)**	**HORAS**	**TOTAL (€)**
Planificación	30	15	450
Investigación y Documentación	30	15	450
DISEÑO DE SOFTWARE			
Desarrollo del software del controlador	30	10	300
TEST			
Pruebas Controlador	30	5	150
COSTE TOTAL DE MANO DE OBRA			1350

3 Coste Total

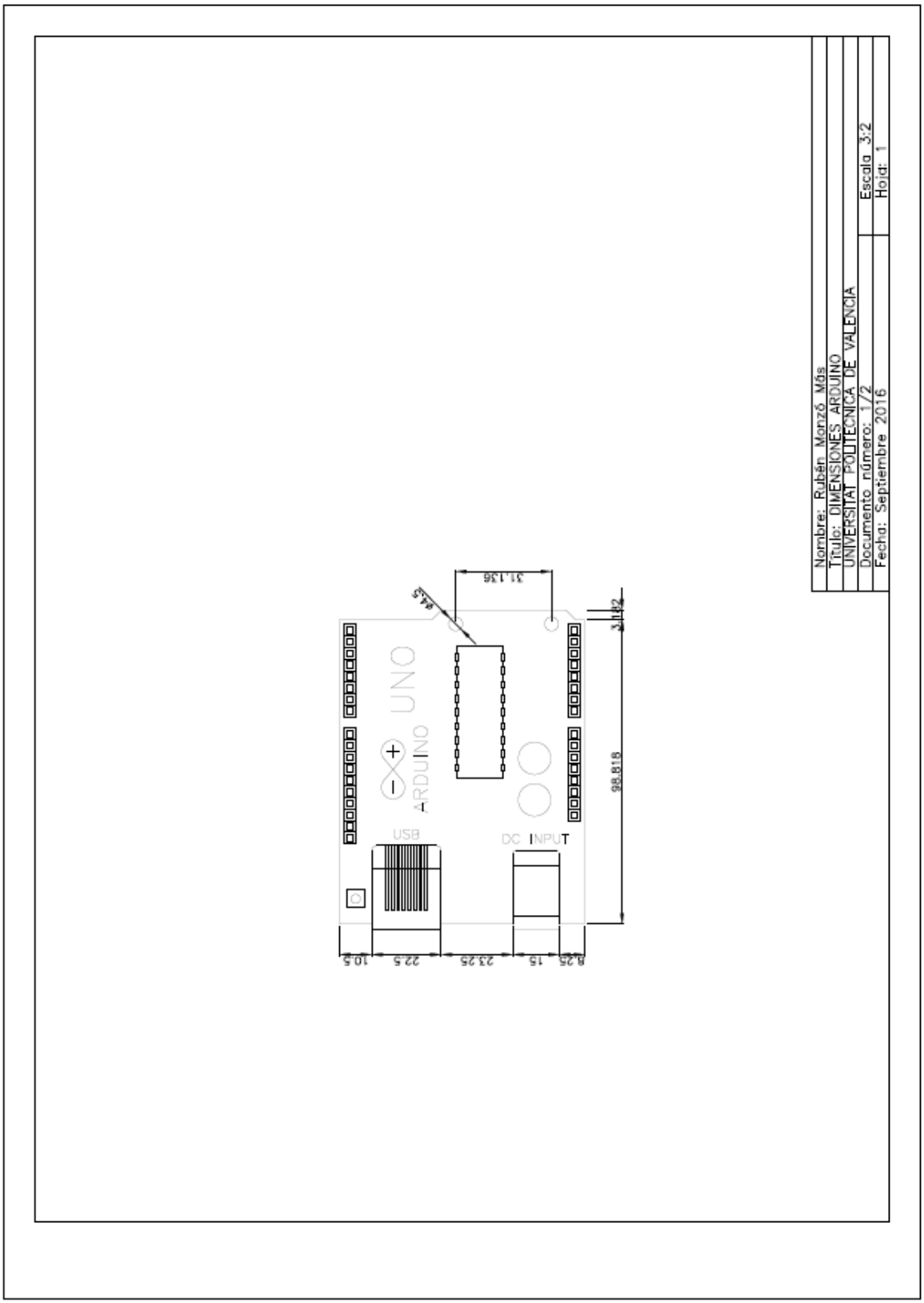
Finalmente, una vez sumados los costes anteriormente calculados, es necesario aplicar el impuesto sobre el valor añadido (IVA).

CONCEPTO	**TOTAL (€)**
Coste de Amortización de Equipos	94,23
Coste de Mano de Obra	1350
Coste Total de Ejecución del Proyecto	1444,23
IVA (21%)	303,29
COSTE TOTAL DEL PROYECTO	**1747,52**

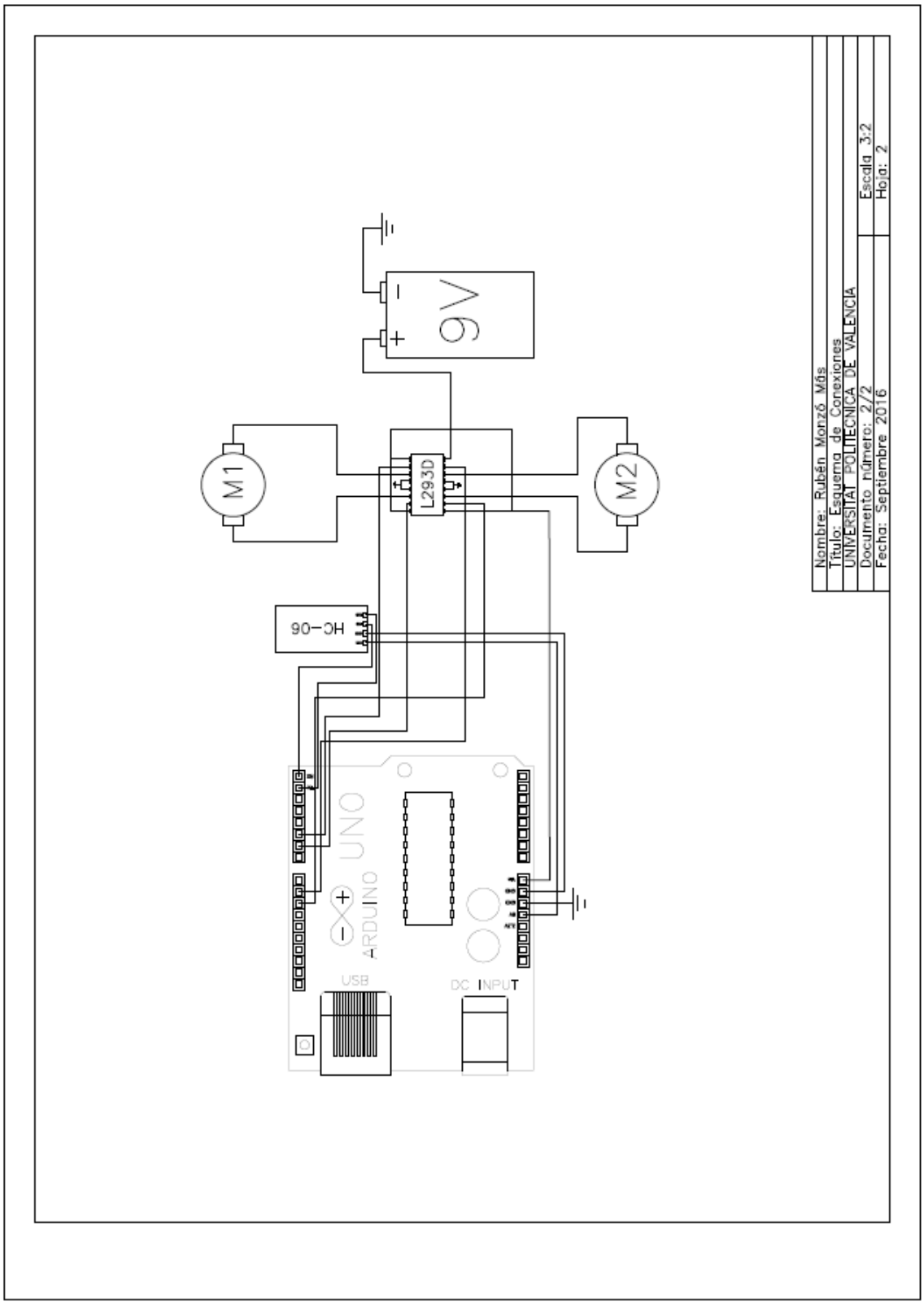
DOCUMENTO II

PLANOS

Plano 1: Dimensiones ARDUINO.


Nombre: Rubén Monzó Más
Título: DIMENSIONES ARDUINO
UNIVERSITAT POLITÈCNICA DE VALÈNCIA
Documento número: 1/2
Escala 3:2
Fecha: Septiembre 2016
Hoja: 1
UNO
ARDUINO
USB
DC INPUT
31.136
3.182
68.818
10.5
22.5
23.25
15
8.25

Plano 2: Esquema del Montaje


9V
M1
M2
L293D
HC-06
UNO
ARDUINO
USB
DC INPUT
Nombre: Rubén Monzó Más
Título: Esquema de Conexiones
UNIVERSITAT POLITECNICA DE VALENCIA
Documento número: 2/2
Escala 3:2
Fecha: Septiembre 2016
Hoja: 2

DOCUMENTO II

PLIEGO DE CONDICIONES

1 Condiciones Generales y Económicas

1.1. Condiciones Generales

Las condiciones y cláusulas que se establecen en este documento son de obligado cumplimiento por las partes contratantes.

I. Tanto el suministrador como el cliente se comprometen desde la fecha de la firma del contrato a llevar a cabo lo que se estipule.

II. Ante cualquier reclamación o discrepancia en lo concerniente al cumplimiento de lo pactado por cualquiera de las partes, una vez agotada toda vía de entendimiento, se tramitará el asunto por la vía de lo legal. El dictamen o sentencia será acatado por ambas partes.

III. Al firmarse el contrato, el suministrador se compromete a facilitar toda la información necesaria para la instalación y buen funcionamiento del sistema, siempre que se le requiera para ello.

IV. Asimismo, el comprador queda obligado a explicar al fabricante todas las características distintivas del sistema en que se va a implantar, con el objeto de facilitar su instalación, quedando el proveedor libre de responsabilidades sobre cualquier deficiencia que surja por el incumplimiento de dicha obligación.

V. El plazo de entrega será de tres meses, a partir de la fecha de la firma del contrato, pudiendo ampliarse en un mes. Cualquier modificación de los plazos deberá contar con el acuerdo de las dos partes.

VI. En caso de retrasos imputables al suministrador, se considerará una indemnización del 1% del valor estipulado por semana de retraso.

VII. Existirá un plazo de garantía de un año a partir de la entrega del sistema. Dicha garantía quedará sin efecto si se demostrase que el sistema ha estado sometido a manipulación o uso indebido.

VIII. Cumplido dicho plazo de garantía, el suministrador queda obligado a la reparación del sistema durante un plazo de cinco años, fuera del cual quedará a su propio criterio atender la petición del cliente.

IX. El suministrador no tendrá en ningún momento obligación alguna frente a desperfectos o averías ocasionadas por un uso indebido por parte de personas no autorizadas.

1.2. Condiciones Económicas

I. Los precios indicados en este proyecto son firmes y sin revisión bajo ningún concepto, siempre y cuando se acepten dentro del periodo de validez del presupuesto que se fija hasta diciembre de 2016.

II. El pago se realizara como sigue:

- 75% a la firma del contrato
- 25% en el momento de la entrega

III. La forma de pago será al contado mediante cheque nominativo o transferencia bancaria. En ningún caso se aceptarán letras de cambio.

IV. El suministrador se hará cargo de los gastos de transporte dentro de la ciudad donde se encuentre la instalación. Si es necesario realizar un transporte interurbano, el gasto correrá por cuenta del cliente. En todo caso, el responsable de los posibles desperfectos ocasionados durante el transporte será el proveedor.

V. Durante el plazo de garantía, la totalidad de los gastos originados por las reparaciones correrán por cuenta del suministrador.

VI. Fuera de dicho plazo y durante los siguientes cinco años, los costes serán fijados mediante acuerdo por ambas partes. Pasado ese tiempo, los fijará exclusivamente el suministrador.

2 Condiciones Técnicas y Particulares

I. El equipo informático deberá estar homologado conforme a las reglamentaciones europea y española vigentes a fecha de diciembre de 2015.

II. El lugar de instalación del equipo deberá ajustarse a los niveles de temperatura y humedad indicados por el fabricante.

III. Los programas informáticos empleados contarán con la licencia preceptiva y se cumplirá con los términos de la misma. En caso de usar programas con licencia GNU, se deberán respetar sus condiciones.

2.1. Hardware

1. Portátil Acer Aspire E15

Procesador: Intel® Core™ i5- 5200U

RAM: 8 GB

Tarjeta gráfica: NVIDIA ® GeForce ® 920M

2. ARDUINO UNO

Peso: 0.025 Kg

Dimensiones: 68.6mm x 53.4mm

Controlador: ATmega328P

2. MODULO HC-06

Peso: 0.004 Kg

Dimensiones: 3.57cm x 1.52cm

Rango alcance: 10m

2.2. Software

- Windows 10 © Microsoft Corporation.
- ARDUINO.
- Microsoft Office Profesional 2010

2.3. Normas de Calidad

Los sistemas se diseñarán de forma que cumplan las normas UNE, CEI y EN aplicables a este tipo de productos.

2.4. Normas de Seguridad e Higiene

El proyecto cumplirá con la Ley 31/1995 de Prevención de Riesgos Laborales.

2.5. Vida útil del producto

Los sistemas deberán tener una vida útil no inferior a diez años en funcionamiento continuo.